USER-ORIENTED INTERACTIVE COMPUTER GRAPHICS

J.C.R. Licklider
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts

This paper deals with several broad idea-issues relating to interactive computer graphics, schematic and pictorial. It considers briefly the trends of the technology and of the market, what the uses and who the users of user-oriented interactive computer graphics will be, and who will do the orienting of the technology toward the users. Then it examines: the concept of stepwise learnability; abstracting graphics functions from graphics applications; the idea of 'dynamic hieroglyphs'; graphics projected onto the retina; graphics embedded in LISP- and APL-like languages; knowledge based graphics; graphic input devices; highly realistic graphics; computer-based arts and crafts; and some possible side effects of advanced applications of computers and graphics.

On the whole, interactive computer graphics appears likely to be one of the main forces that will bring computers directly into the lives of very large numbers of people during the next two or three decades. Truly user-oriented graphics of sufficient power to be useful to large numbers of people has not been widely affordable, but it will soon become so, and, when it does, the appropriateness and quality of the products offered will to a large extent determine the future of computers as intellectual aids and partners of people.

Introduction

This session is on user-oriented interactive computer graphics. In terms of this four-fold table, the session is on user orientation of the right-hand column:

	Non-interactive	Interactive
Schematic	e.g., point-and-line graphics on paper & film	dynamic and viewer-controlled point-and-line graphics
Pictorial	e.g., Utah's pseudo-photographic pictures	yet to come

In the table, "schematic" means "composed of points and lines" or otherwise heavily abstracted from the full, detailed, pictorial situation or representation, and "pictorial" means (perhaps somewhat arbitrarily) "retaining the full detail, as in a high-resolution photograph or realistic scale model". It seems important to me that the kind of graphics identified by the lower half of the right-hand column does not exist yet -- but can be anticipated within this decade or the next and can be expected to appeal to vast numbers of people who have little interest in schematics. Indeed, it seems important to me that we are in a field that, although about a quarter of a century old, is just now acquiring the kind of technological base it needs in order to take off and fly!

The advance of LSI technology -- especially the progressive increase in the density of logic and memory elements on chips and the progressive decrease in cost per element, which are following steeper exponentials than the traditional "double or halve every two years" that has characterized computer hardware parameters since 1943 -- is rapidly providing the storage and processing facilities that interactive computer graphics requires. The display capability is still provided mainly by cathode-ray tubes (CRTs), and with reasonable effectiveness. Indeed, cathode-ray tubes are proving to have even more staying power than core memories. However, there is a growing need for a medium that will do for display what LSI chips are doing for processing and storage. Such a medium seems not to have presented itself thus far -- unless it is the plasma panel, and unless the limited resolution of the plasma panel is just a "childhood phase". On the hardware side, what user-oriented interactive computer graphics needs most is an inexpensive high-resolution surface that can be written upon and read from by both computers and people, that is persistent but selectively erasable, and that provides high brightness and a full range of color. Every desk should have such a surface. Then we should set up a cry for three-dimensional display media.

It seems likely that major advances in computer graphics and in computer networking will take place concurrently during the next ten or fifteen years. Networking will greatly complicate interactive graphics systems -- will probably require that remote graphics consoles be equipped with considerable processing power and memory capability -- because interactive graphics systems are quite sensitive to time delays and bandwidth restrictions. At the same time, networking will greatly increase the number of potential users of interactive graphics systems.

The trend of the marketplace should be observed, by way of introduction, along with the trend of the technology. Averaging a number of pertinent assertions by industry leaders, I come to the conclusion that consoles (terminals) are decreasing in price about 25 percent per year, that in terms of total dollar sales the console market is increasing about 25 percent per year, and that cathode-ray displays now predominate over typewriters and printers. However, point-and-line graphics is still the limited-volume top-of-the-line Cadillac or Continental, as compared with straight Chevrolet/Ford alphanumerics, and highly capable graphics systems have the same share of the market as Mercedes plus Rolls Royce. The thought that keeps the analogy from depressing me is the initial thought of the preceding paragraph: the way things are going, although almost no one will be able to afford a car with four wheels in the year 2000, nearly everyone will be able to afford a highly capable graphics system.

What, then, about "user orientation"? What will the uses be? Who will the users be? Who will the orienters be? What will "user-oriented" really mean? Let me deal with the first three of those questions only very briefly as I conclude this Introduction. Then let me discuss, in an attempt to deal with the fourth, ten facets of user-oriented computer graphics that seem important to me and that I want to recommend to you.

What will the uses be? First, "computerization" and "interactivization" of many functions now served by print-and-paper technology. Second, computer based education, going far beyond both print-and-paper technology and present-day computer assisted instruction. Third, entertainment, which may well be the largest and most profitable application area for computer graphics. Fourth, new ways of perceiving, learning, knowing, solving, and creating -- a veritable cognitive revolution that may begin as soon as there is plentiful processing power and memory capacity and school children discover that interactive computing is much more fun than watching television.

Who will the users be? In a phrase, they will be the menials, the mentals, and the masses. There will continue to be many workers in business and industry and in government who use computers according to simple rules, following prescribed procedures, impedance-matching computers into the flow of decisions and events -- and more and more they will work with pictures in addition to symbolic identifiers of things. There will be increasing numbers of designers, problem solvers, and decision makers who use computers directly -- and deeply -- in their work, and most of them will use interactive graphics as well as other cognition-supporting tools. And, perhaps most importantly, in the long run, computers will directly enter the lives of vast numbers of ordinary people. Thus the great majority of users will not be programmers, not typists, not computer or information specialists of any kind. The effectiveness with which they use computers may depend heavily upon how effectively the graphical interaction techniques are oriented toward their needs and capabilities.

Who will the orienters be? Those who do the orienting, who design and program computer systems for the three sets of users will be: for the menials, computer professionals; for the masses, computer professionals; for the mentals, in part professionals and in part the users themselves. For the first two sets of users, the various application systems will be designed and delivered as completed packages. For the third set, the range of applications will be too great for completion before delivery, and many creative users will extend the systems with which they work. That projection adds an important dimension to user orientation. It will be necessary for the designer to view the prospective user as a design extender and the delivered system not only as an application package but as a kit of design-extension and incremental-development tools.

Stepwise Learnability

Because graphics opens wide new horizons to designers and users of interactive graphic systems, interactive graphics systems tend to be both novel and complex. Typically, the user has to do a lot of learning in order to become proficient.

It would be wrong to argue against either novelty or complexity *per se*. It is not good to do something *in a* new and unfamiliar way unless it is better than old, familiar ways, but graphical techniques which will be novel to most users during the coming decade, offer so many advantages and superiorities over nongraphical techniques that they can repay the cost of a little learning time. Unwarranted complexity is bad, but many of the tasks that graphics will bring into the range of feasibility are inherently complex, and it will often be better to master a complex procedure and develop a complex skill than to leave the task undone. What is needed is a way to make new and complex techniques readily learnable.

The approach to that end that I want to recommend is to design interactive graphics systems in such a way that new users can master them in a series of small steps. At the end of the first session -- the first half-hour alone at the console or the first two hours of a tutorial -- the neophyte should be able to handle some significant part of his task with enough competence and dispatch to let him do that part on-line thereafter. Then it should be very quick, easy, and rewarding for him to learn each successive step and incorporate it into routine work procedure. Whereas a tutorial session may often be afforded for a brief introduction, only rarely will the new graphics user be able to go off to school for a course. Most of the advancement to the level of expert will be achieved through on-the-job learning, and such learning can occur efficiently only if it grows out of the design of the system being learned, out of its essential modularity and its naturally stepwise progression of functionality, language, and procedure.

Abstracting Graphic Forms and Graphics Operators from Applications, Tasks, and Systems

The great advantage of pictures over words lies in their higher dimensionality and greater potential variety, but most computer graphics systems offer too much freedom. One must impose constraints before he can achieve either art or communication. The most fundamental part of user orientation is the one involved in determining the basic constraints that will define and structure the repertoire of graphic forms and graphics operations, that will limit it to a learnable and usable size.

The general problem is familiar from experience with assembly languages and higher-order programming languages. Although the latter retain full "Turing capability", they greatly decrease the number of trivially different ways one can represent each essential datum or implement each essential algorithm -- and at the same time decrease the amount of text required to express it. In the development of programming languages, it has turned out that a set of a dozen or so basic data structures has turned up over and over in the many and diverse application areas that computers have invaded, but that, even though simple mathematical functions are encountered repeatedly, they do only a fraction of what must be done, and one must have a library of several thousand operators (functions or procedures) before he can handle diverse applications without writing many *ad hoc* subprograms for each of them. In *interactive* computer graphics, there has not yet been a sufficiently comprehensive analysis of the forms and operators essentially involved in a wide variety of applications. Such an analysis must be made to define a basic set of formats and functions that users can master once and use in many contexts.

If we were concerned only with static graphics, one display at a time, I would expect analysis to come up with the same basic data structures that already are familiar from the high-level programming languages, but each would have to be displayable in several different formats, and several basic structures would often be combined in a single presentation. Dynamic graphics, color, motion, and multi-display consoles will complicate the picture further. Users, however, will not be able to tolerate the level of complexity those sentences suggest. I think that this may be the most important single problem in interactive computer graphics: to abstract the bare essential repertoire of forms and operations from a diversity of applications and to establish a standard protocol for implementing them in systems -- a protocol including display parameters, control parameters, and nomenclature as seen by users.

Dynamic Hieroglyphs

Lest you get the impression that I am surer than I am about the importance of making the situation as simple for users as is consistent with the task to be facilitated, let me mention an idea at the other end of the complexity scale that nevertheless appeals to me greatly. It starts out with the observation that, although alphabetic (or alphanumeric) languages with fewer than a hundred printing characters appear to have won out in the modern western world, there is something to be said for very much larger "character sets". In general, the more characters or other elementary signs, the shorter the representation of an idea. Chinese uses much less space per idea than English, which -- with its larger vocabulary of words -- is more compact than the other alphabetic languages. Mathematicians are always running out of Latin characters and adding Greek, Latin, Cyrillic, Hebrew, and German Script to their repertoire. Now that we have the computer to serve as vocabulary drill master and quickly give us very large vocabularies, why not go back to hieroglyphs and work with an ensemble of several thousand, rather than only several dozen, basic signs?

Why not, indeed, let the basic signs be kinematic? Each "dynamic hieroglyph" could then truly "tell a story". A dynamic display of dynamic hieroglyphs could represent hundreds or thousands of interactions among tens or hundreds of ideas. To the untrained eye such a display would no doubt look like a bucket of worms, but to the trained eye...? Would it exceed fundamental limits of human cognition? Or would it merely require thousands of hours of training (as a chessboard does) to be perceived as an organized, coherent whole?

Dynamic hieroglyphs would have to overcome two obstacles before they could revolutionize interactive computer graphics. They would have to work, and then many people would have to devote major segments of their lives to learning them. Abstracted graphic forms and graphics operations will surely work, and the average user should master them in a couple of hours. A practical evaluation therefore will assign higher priority to abstracting the forms and operations, but dynamic hieroglyphs appeal to me greatly.

Graphics Projected onto the Retina

You may think that I am interpreting "user-oriented" in an overly literal sense, but I am willing to go to any length to rationalize the inclusion of this idea in my list of ten ideas. The essence of the idea is that graphics does not fundamentally require any other display screen than the retina of the eye. The essential task of a graphics system is to project properly selected photons onto properly selected retinal locations at properly selected times. A secondary component of the idea is that feedback from the user or from sensors located or trained upon his eyes or head or body may be useful in making the proper selections.

Most of you will remember Ivan Sutherland's retinal projection system employing a head-mounted cathode-ray tube, lenses, and a semi-silvered mirror, together with a data take-off that told the computer the orientation of the viewer's head as a function of time [1]. The computer had in its memory a model of some situation -- actual or hypothetical -- with the viewer in the middle of it. The computer always displayed (i.e., projected onto the real viewer's retinas) what the correspondingly oriented model viewer was "seeing" in the modeled situation. In Ivan's laboratory at Harvard, I had the experience of walking around in a model-perceptual room (inside the larger real laboratory room), a room made out of lines, as in a mechanical engineering drawing. Later, in his laboratory at Utah, I had the dubious pleasure of visually steering a missile from a point in space from which I could see most of North America toward an intended point of impact, the Rand Corporation in Santa Monica, California. I was in the nose of the missile, and the missile went where I looked. Fortunately, I blinked just before impact, and the missile hit about half-way to Catalina Island. Rand is still in operation. But I retain some very vivid imagery, together with the conviction that an important part of user orientation involves taking advantage of the user's best built-in or already mastered skills -- such as tracking a target by eye without involving the hands.

A few years ago, NASA developed an eye-tracking device that observed a human eye from a distance and figured out the direction the eye was looking. Marvin Minsky borrowed the device, and one of his students, Sam Geffner, programmed a demonstration that embodies another neat bit of user orientation [2]. What does a person who is reading a foreign language do when he encounters a word he cannot translate? He pauses, and so do his eyes. The computer displayed sentences of French in letters large enough to permit the NASA device (with the aid of some calculation) to determine where, in the text, each eye fixation fell. Whenever the fixation point lingered too long in the vicinity of a word or phrase, the computer substituted the (pretranslated) English equivalent for the French word or phrase. The demonstration was just a concept feasibility demonstration--the viewer's head was clamped in a jig--but the idea impressed me strongly. The essential point, of course, is to let the computer know what the user is doing so it can help him. I wonder whether a clue to or harbinger of the condition, "I don't know the translation", can be derived from EEG waves. Of course, I personally would be willing to press a button.

Graphics Embedded in LISP- and APL-Like Languages

Graphics has dominated most of the computer systems (such as systems for computer aided design and systems for the layout of LSI chips) that have made extensive use of interactive graphics, and the languages through which users have interacted with such systems have been primarily graphical languages. The languages in which such systems have been implemented, however, have usually been ordinary programming languages with little or no graphics capability of their own. It seems to me that there is an important middle ground, a place for systems that employ both graphical and alphanumeric input and output and that are operated or controlled through mixed media. It seems to me, also, that there is a need for languages that can serve both as programming and interaction languages and that incorporate the essential graphic forms and graphics operations along with the data structures, control structures, and basic operations required in programming systems and in interacting with systems that have been programmed.

My favorite candidate language for the role just described is a derivative of LISP called "MDL" ("More Data-structures than Lisp"). But instead of pushing it, particularly, I want to point in the general direction of LISP and APL, languages that have implementations with both interpreters and compilers, the interpreters being in fact complete programming systems with built in libraries, documentation subsystems, debugging aids, and so on -- and the compilers being already reasonably efficient in some instances and in principle very amenable to high optimization. I think that some characteristics of LISP and APL are of the essence of user orientation, and I think that interactive computer graphics can correct some of the characteristics that are not. Some of the good features of LISP are the simplicity of its data and control structures, the essential modularity of LISP programs (functions), the fact that LISP programs are processible as data by other LISP programs, and the fact that a single program can include both interpreted and compiled parts. Some of the good features of APL are its abundance of array operators and the terseness with which complex procedures can be expressed. A few bad features should be acknowledged: LISP has too many parentheses and too many terms that are not properly self-explanatory, and APL reads backwards, but graphical representation could cure those ills.

What I advocate, then, is a graphics-augmented LISP-like or APL-like language that can be used both to program and to operate interactive graphics systems. For nonprogrammer users, some of the more sophisticated parts of the language would ordinarily be held back, but essentially there would be a single, uniform language for graphics and nongraphics, for programming and operation. The main consideration that forces me to qualify my claims for the value of such a language is that both LISP and APL complications require much further development before they can provide the efficient code required in the final stages of system development. However, as Bruce Daniels is arguing very convincingly [3], the greatest optimization can be achieved with the highest-level language, and the future therefore looks good for efficiently compiled code from such a language as I am advocating.

Knowledge Based Graphics and Visualization

In the development of computer graphics, one very important area seems to be lagging the other areas rather markedly -- and remarkably. It is the area that is inverse to abstraction. Let me call it "visualization". The basic notion is that the essential features of a complex structure or process can be abstracted from the residual detail and represented succinctly and economically for many uses, but, when it comes time for realistic presentation of the structure or process, the detail has to be put back in -- or, as in an architectural drawing of a brick building, at least artfully suggested. Both the analytic process of abstraction and the synthetic process of disabstraction -- visualization -- are knowledge-based processes. In the field of artificial intelligence, a considerable amount of work is being done on abstraction. It is visualization that I think is being neglected. A natural home for work on visualization is computer graphics.

Visualization can be attacked in the same way as programming is being attacked in the most advanced software centers. In the case of sophisticated programming, there is a point at which the essence of an algorithm has been expressed in a very abstract, high-level language, and the problem is to fill in the detail required to permit the algorithm to run in an actual, specific computer. A corpus of knowledge about programming in general, about the high-level language, and about the target machine is represented in a computer-processible knowledge base associated with the "intelligent compiler" or "automatic programmer". The translation from the abstract level down to machine language is carried out with the aid of continual reference to the knowledge base. In analogy, I envision a graphics translator or compiler and a knowledge base that "knows a lot about" (for example) steelwork, welding, and riveting. In an extreme case, the abstract representation may specify only the center lines of the columns, beams, and girders of a skyscraper and whether they are welded or riveted together, but the knowledge base knows -- or the program can figure out from information it gets from the knowledge base -- the cross sectional sizes and shapes of the members, and it can display the detail view of a given corner, even down to the irregularities of a weld and the orange paint.

Indeed, as I visualize the example, I see that the visual graphics system in my own head does fairly well, though with less than real perceptual clarity and vividness, what I am suggesting computer graphics systems should do, and do very clearly and vividly. In this connection, user orientation will involve acquiring and organizing the knowledge of the users' fields that will be required to fill in the detail that is implied but not expressed by the abstracted essence, and user orientation will involve mastering the techniques of the artist and the architect -- techniques of the kind Nick Negroponte and his associates have been studying and transferring to computer science -- that make it possible to make the eye see a meadow without displaying a trillion blades of grass.

Light Pens and Stylus Tablets

Lest I wholly neglect the field of graphics input, let me mention very briefly the two devices that seem to me to tie for the booby prize in user orientation: the light pen and the stylus tablet.

The light pen evolved from the light gun, a pistol-like device of the early 1950's, intended to identify a "blip" on a computer-based radar display. The operator aimed the barrel at the place where the blip was expected to appear and pulled the trigger. If the blip appeared and was "seen" by the photocell in the barrel, the gun sent the computer a pulse, and the computer could figure out the range and azimuth of the target from the scan pattern. But the barrel of the light gun was usually too fat, and the end of the barrel so occluded the blip that the operator often missed it when it appeared or thought he hit it when it did not appear. Light pens inherited the obesity of light guns. Even worse, the displays with which they were used were almost always oriented vertically. The result was to tire the users' arms, to fail to take advantage of their years of experience with pens and pencils on horizontal surfaces, and to doom the light pen to an early grave.

The demise of the light pen made way for the stylus tablet. The stylus was properly small and light, and the tablet on which it "wrote" was properly horizontal. But the tablet was not the display! In almost every stylus-tablet graphics system, the display was (or is) a vertical CRT screen, and the tablet competed with a keyboard for the horizontal working area at the base of the screen. Many times I have heard that layout defended on the ground that users quickly learn to operate the stylus while looking at the screen, but in fact the dissociation of the tablet from the display screen impairs performance significantly in some of the most basic graphic-input tasks. For example, place a map of the United States on a horizontal surface and touch the state capitals one after another with a pencil or stylus. Then display the map on a CRT screen and touch them with a cursor controlled by a stylus and tablet. The latter series will take twice as long as the former even after you have had several hours of practice with the stylus.

Obviously, the solution is to put the display screen down on the desk top and use it as a common medium for both the computer and the user. Mort Bernstein and Lou Gallenson did that many years ago in a beautiful concept-feasibility demonstration [4]. The computer turned the user's imperfect block letters to precise printing, straightened lines, squared up corners, and so on, right before his eyes, right when and where he wrote and drew. However, user orientation is not just a matter of putting the displays and the controls in the right places. Mort's system required a lot of computer power at a time when it was very expensive; it required a large and expensive cathode-ray tube; and it required a darkened room. Happily, the advance of the technology is overcoming those obstacles, and soon we can have user-oriented graphic input -- if we haven't gotten so accustomed to stylus tablets that we fail to notice how bizarre they really are.

Realistic Graphics and Computer Power to the People

For users who are good at working with abstractions, the preferred graphic forms are likely to be sketches, drawings, charts, diagrams, graphs and the like. For the average man, however, it is necessary either to educate and train him in the use of abstractions or to give him realistic graphics. By realistic graphics I mean essentially pictorial graphics with full gray scale, shaded surfaces and shadows, as developed by General Electric and Evans and Sutherland -- but at a much lower cost than the system GE built for NASA and a much lower cost than the LDS-1 and LDS-2. Color and dynamics are useful, of course, in both abstract and realistic graphics.

Although there will be a lot of education and training in abstraction during the coming years, some of it via electronic TV games and pocket computers with graphic display panels, I think that it will probably be realistic graphics that dominates the mass introduction of computing into the lives of the people. To what extent the graphics and the computing will be interactive, I am not so sure, but I strongly believe that, when there is a computerized entertainment/information center in every home, maps will be picture maps and computer generated representations of automobiles and airplanes and buildings will look like photographs of automobiles and airplanes and buildings. In any event, this is one of the major questions in user orientation because in the future the most numerous users will be not specially trained information processors but ordinary people. Will they require realistic graphics? High resolution? Tight interaction? Or just points and lines, TV scan, and push buttons?

Computer-Based Arts and Crafts

Less numerous than "ordinary citizens" but nonetheless looming large in my projection of computer graphics into the end of the century are artists and craftsmen, both professional and amateur. Computers may become important to them in two ways, both essentially involving graphics.

First, the artist or craftsman may work through a computer into his traditional medium. One can imagine a sculptor, working with a Roberts Wand instead of a chisel, shaping a graphically displayed computer-based model instead of a block of stone [5]. He would work on the model until satisfied with it, then command the marble-cutting subsystem to translate the model into more substantial form. He might then return to the computer-based model, revise it, and then see how the revised version looked in stone. (One can imagine, also, that many artists and craftsmen will resist or react against such a departure from traditional procedures. Is the important thing to achieve the envisioned form, to achieve the envisioned form in stone, or to achieve the envisioned form in stone with a hammer and a chisel?) The basic notion is applicable, of course, to painting as well as sculpture -- and to weaving (cf. Janice Lowrie's work with a computer-controlled loom at IBM [6]), jewelry making, and many other arts and crafts. It has obvious implications for the reproduction of objects of art. It may one day be an important focus for user orientation of computer graphics.

The second way computer graphics may affect art and craftsmanship is to become, itself, an accepted medium of artistic expression. In the terms of the preceding paragraph, if the computer can hold the sculpture in its memory and display it in three dimensions continually or upon request, why translate the model into stone? Clearly, the computer medium offers more dimensions, more opportunities -- for example, freely kinematic sculpture -- and permits the definition of exactly those constraints, and only those constraints, required to meet the artist's needs. (No unsupported masses. No cracks in the stone!) Of course, there are two sides to all of this, and I do not mean to urge computerization against the sensibilities of artistic tradition -- I merely want to point out what all programmers know but only a few others have yet discovered : that the computer medium is the nearest thing yet to an ideal medium for the representation of ideas. Computer graphics is not yet an ideal medium for the expression of ideas. It may be important to the future to make it so.

Prevention of Dysfunction

For the first three decades of the computer era, the main problem of system design was perceived to be the provision of function. As long as a system did effectively and efficiently what it was supposed to do, it was considered to be successful. Recently, however, the importance of the other side of the coin of function has begun to be recognized. It is no longer sufficient to provide function: it is necessary also to prevent dysfunction. The designer has to think, for example, of privacy, security, back-up facilities, and the effect of reducing human contact with clients or customers. These are just as much matters of user orientation as are considerations of display brightness and interaction language. It is just that the users in the focus of concern are less likely to be the ones sitting at operators' consoles.

It is unfortunate, in a way, that design problems become complicated by concern about dysfunction just as interactive computer graphics begins to enter its ascendancy. Design problems were difficult enough already. Nevertheless, the concern is here. A long-persistent display screen may be very helpful in presenting the complex engineering drawing of a highly proprietary new product, but the display should not still be readable ten minutes after tha authorized viewer has logged out, locked up his safe, and gone home. The laser that writes so beautifully on the semiconductor display surface may leave a small blind spot in the console operator's fovea. The ultrasonic beams of a Roberts Wand may transmit the graphic input signals to an eavesdropper in the next room as well as to the computer. So it is essential to take into account all the user's needs and concerns, and that makes user orientation a more difficult game.

References

1. Sutherland, I. "A Head-Mounted Three-Dimensional Display", Proc. Fall Joint Computer Conf., 1968.

2. Geffner, S.L. An Eye Tracking Program, M.S. Thesis, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Jam. 1968.

3. Daniels, B. Very High Level Optimization: Design of an Optimizer for APL, Ph.D. Thesis Proposal, Laboratory for Computer Science, Massachusetts Institute of Technology, Nov. 1976.

4. Gallenson, L. "A Graphic Tablet Display Console for Use Under Time-Sharing", AFIPS Conf. Proc., Fall Joint Computer Conf., Vol. 31, 1967, pp. 689-695.

5. Roberts, L.G. "The Lincoln Wand", AFIPS Conf. Proc., Fall Joint Computer Conf., Vol. 29, 1966, pp. 223-227.

6. Lowrie, J.R., Lorenzo, J.J., and Bomberault, A. "On-Line Textile Designing", Proc. 21st Nat. Conf., Assoc. Computing Machinery, 1966, pp. 537-544.